Washington University in St.Louis

OLIN BUSINESS SCHOOL

FL2022.B69.DAT.560M

Big Data and Cloud Computing

Final Report

DAT 560M Sec 29

Prof. Hossein Amini

Date: 08th, Dec.2022

Group 60: Ruichun Li (515524), Anyi Zhang (515522)

Zixiao Wu (515491), Enoc Yao (515500)

Executive Summary

The purpose of this report is to figure out the keys to financial success in the movie industry. The goal is to draw insights through "The Movies Dataset" based on Kaggle dataset using big data method, our team stated the problem of financial success in the movie industry, explored the possible features that may affect movies' financials, and applied different analytical tools to process data (**PySpark**) and visualize data (**Matplotlib** and Tableau). As a result, we discovered top genres that might generate the highest ROI (return on investment) and how movies' popularity, runtime and rating would impact the ROI. In light of our analysis, we offered three recommendations for improving movie industry financial success.

Description of the data

The dataset is called "The Movies Dataset". It contains metadata on over 45,000 movies. 26 million ratings from over 270,000 users. This data was collected by Rounak Banik who ensembled the data from TMDB and GroupLens in 2017. The size of the dataset is 943.76 MB and has 45,572 records. This structured database includes cast, crew, plot keywords, budget, revenue, posters, release dates, languages, production companies, countries, TMDB vote counts and vote averages in 7 files and 43 columns.

Why this is big data

This dataset provides comprehensive information about the movie industry that allows our team to conduct a thorough analysis. According to the three Vs rule, this data is considered big data. Big data is characterized by greater variety, larger volumes, and faster speeds.

- **Variety:** It contains many types of data that are available.
- **Volumes**: It has high volumes of low-density, unstructured data
- **Velocity:** It possesses a fast rate at which data is received and acted on.

Problem Statement

The public can be fickle, and the industry is in flux. Just about any movie is an extremely risky investment, even a film starring big-name actors and actresses. The profitability of a film has always been an important reference for investment in film production. If a film does not make a return or even a profit, the interests of the film investors cannot be protected. The profitability of a film is also directly related to the prospects of the production company. If a film is not guaranteed to be profitable, a studio may face the dilemma that no one will invest in it.

Based on the TMDB Movie Dataset, our purpose is to examine what characteristics should be common to movies that are financially successful? In other words, what factors correlate with the profitability of a film at a statistical level? So, this report explores the dataset and answer various questions about the factors that lead to a movie's financial success. From the database, we selected some representative movie features to ask questions and examine:

- Is there a correlation between the genre of the film and the profitability of the film?
- Is there a correlation between the popularity of the film and the profitability of the film?
- Is there a correlation between the runtime of the film and the profitability of the film?
- Is there a correlation between the rate of the film and the profitability of the film?

By examining these questions, we can get a rough overview of the characteristics of profitable films, and furthermore, we can make a brief consideration and analysis of the logical relationship between these factors and film profitability.

Method

The analysis process can be divided into two parts: data processing and data visualization.

In terms of data processing, **PySpark** is used in the following steps:

1. Join movies_metadata.csv and rating.csv together as our dataset.
2. Add the column 'ROI' to the table using formula (revenue-budget)/budget and column 'revenue' and column 'budget'.
3. Figure out the relationship between genre and financial return.
    - Redefine the column 'genre'.
    - Calculate the avg of ROI, Revenue and Budget with genre group.
4. Calculate correlation coefficient between Popularity, Runtime and Rate.
5. Figure out the relationship between Popularity, Runtime, Rate and ROI.

In terms of data visualization, **Matplotlib** and **Tableau** are used to create bar plot and scatter plot.

Results

Genre and ROI

In line with reality, Animation, Fantasy, and Adventure have the highest revenue. However, they are not among the genres with top ROI. Interestingly, Documentary, Horror and Music have the highest ROI. This is because those high grossing movies are accompanied by high investment. As we can see from the table, Animation, Fantasy, and Adventure have the top budget. But a genre such as Documentary has a low budget, which probably leads to high return on investment.

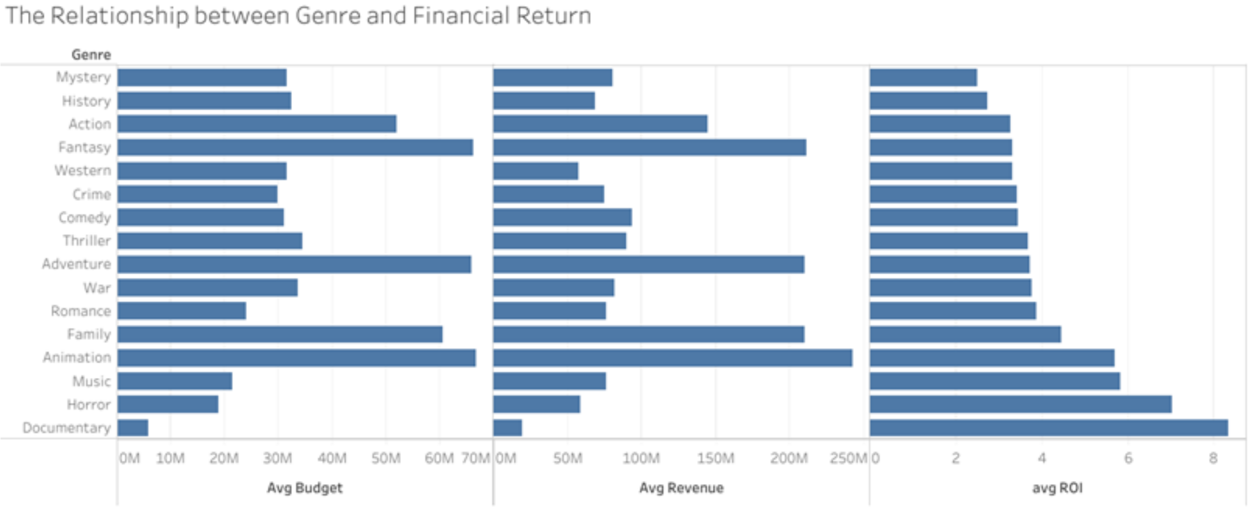
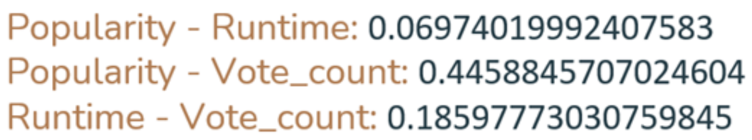


Figure 1 Genre-Revenue & ROI Relationship

Correlation Coefficient

As is calculated, we found that there is a correlation between popularity and vote count(rate). That means popularity, the influence of the movie before release can have influence on the rating and have further influence on ROI, the financial success of the movie. Besides, the correlation coefficient between popularity and vote count is positive, which means the relationship is positive.



```
Popularity - Runtime: 0.06974019992407583
Popularity - Vote_count: 0.4458845707024604
Runtime - Vote_count: 0.18597773030759845
```

Figure 2 Correlation Coefficient

Popularity and ROI

Popularity refers to the influence of a movie before it is released. It is hard to define the relationship between popularity and ROI. Although the dot plot shows that the less popular movie sometimes has higher ROI, most of the less popular movies suffer from relatively low ROI. And movies which do not have a high popularity may have the risk of getting negative ROI, which means losing money.

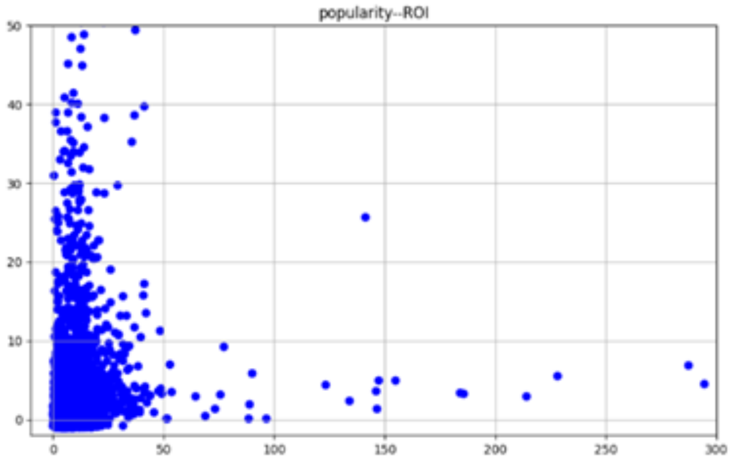


Figure 3 Popularity-ROI Relationship

Runtime and ROI

The scatter plot shows that the runtime of movies with the highest ROI is concentrated around 100 minutes. And movies which run longer than 200 minutes usually get low ROI.

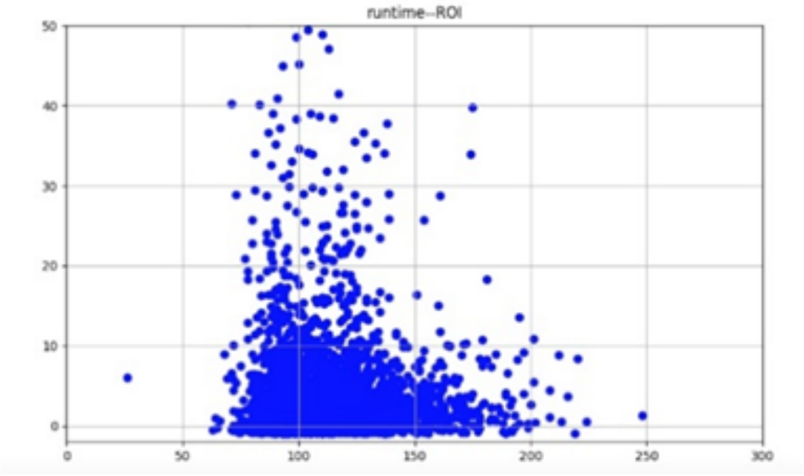


Figure 4 Runtime-ROI Relationship

Rate (vote > 100) and ROI

According to the Rate-ROI relationship plot, movies with higher rate are more likely to have the chance to have higher ROI. And low-rate should be avoided because low rate movies always have low ROI.

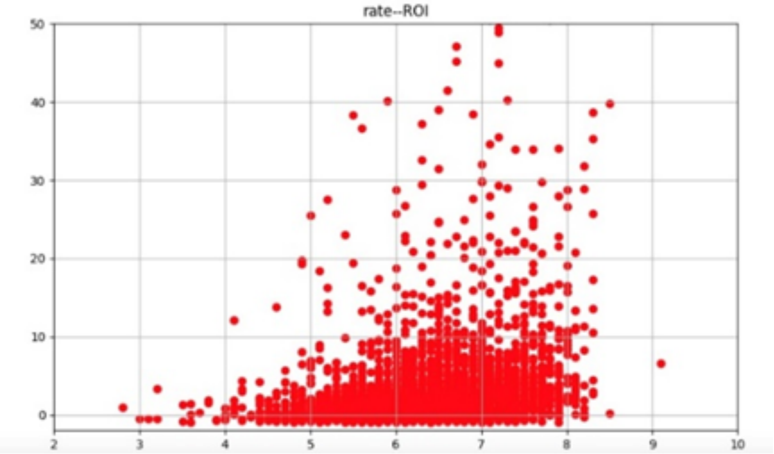


Figure 5 Rate-ROI Relationship

Conclusion

In this work, we considered 4 features that may affect movies' financial success. We use ROI(return on investment) as an estimate of financial success and the 4 features are *Genre*, *Popularity*, *Runtime* and *Rate*.

The first outcome is that documentary, horror and music are three genres having the highest ROI. These genres are the ones that can create brand new hotspots and watching experiences, and the total cost is relatively low for mostly they rely on editors to set brilliant movies topics and the actual content.

For popularity, it has no clear trend related to ROI. The popularity can be interpreted as the fame of the movie starting from the very beginning of the movie promotion stage. It's found that the popularity of the most blockbust movies are quite low(0-10) compared to some between 50-60. Popularity won't guarantee financial success.

As for movie runtime, length of approximately 100 minutes performs the best. If the movie is too long, it's hard for the audience to comprehend. And if it's too short, like 80 minutes, it's on the contrary difficult to tell a good story.

When it comes to movie's rate, the higher the rate, the higher the ROI. It's a virtuous circle that more people watch the movie and when votes increase.

Recommendation 1:

There’s no need to chase current hotspots as current popularity is not a decisive factor for financial success, and it’s the same for making series movies. Movie makers should make their own hotspots.

Recommendation 2:

Avoid lengthy or too short narrative structure. Story is the foundation of one movie, appropriate and attractive narrative structure is very important for watching experience.

Recommendation 3:

As previously discussed, there is a quite high correlation between popularity and movie rate. The relation between two of them may be nonlinear, but popularity does affect movie rate, which to a large extent decides financial success. Accordingly, movie promotion plays a certain role in its consecutive financial success, and movie makers should pay some attention to the promotion stage.

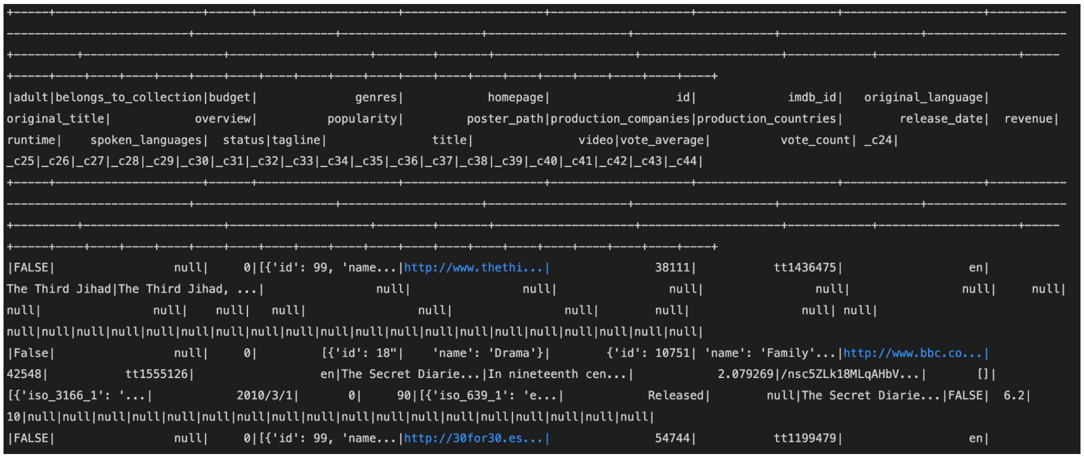
Appendix

Dataset: The Movie Dataset
https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset?datasetId=3405&select=keywords.csv

Codes:

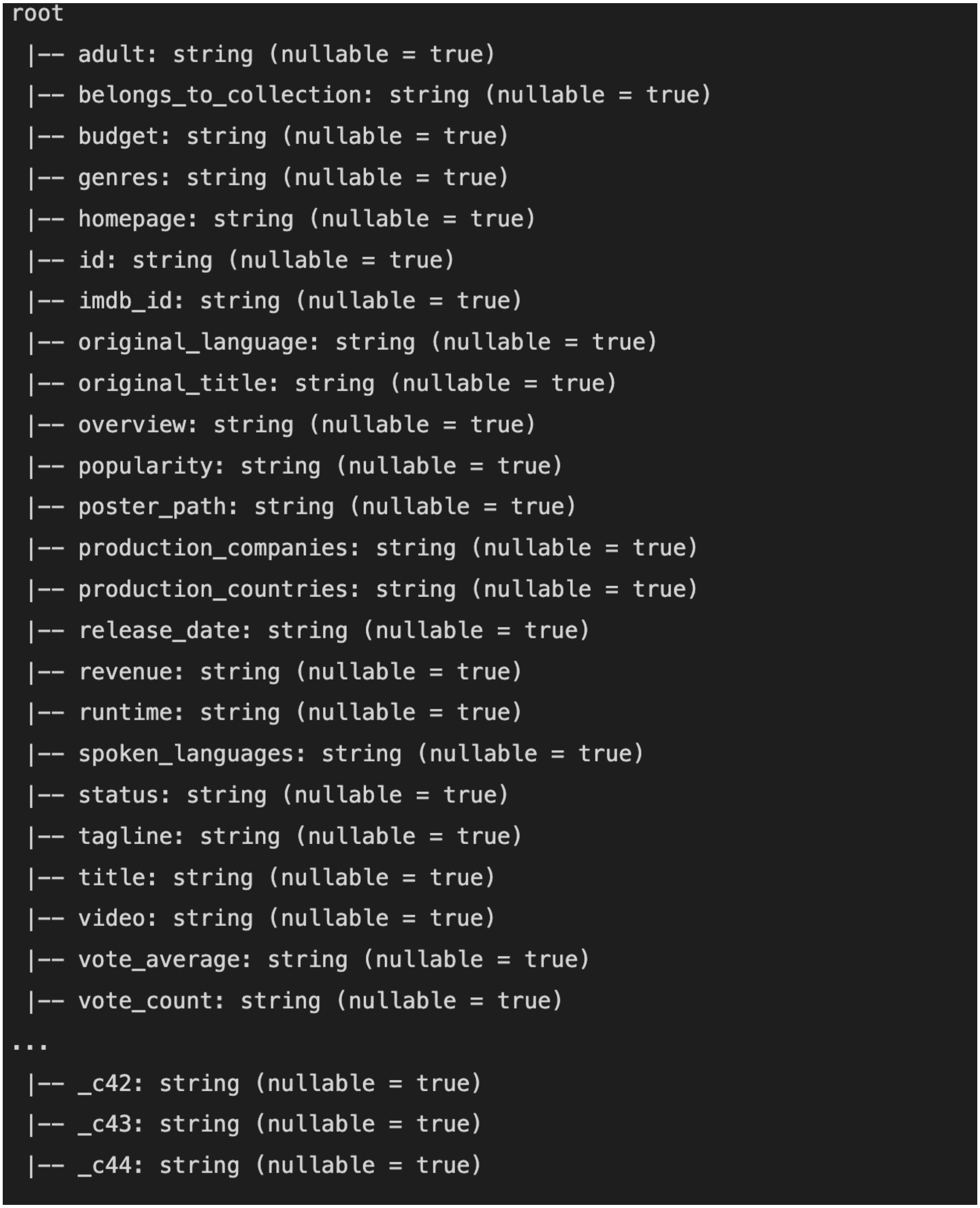
Data manipulation

```
import findspark
findspark.init()
import pyspark
from pyspark.sql import SparkSession
spark=SparkSession.builder.master("local").appName('final').getOrCreate()
sc=spark.sparkContext

%matplotlib inline
import pandas as pd
import matplotlib.pyplot as plt

from pandas.core.frame import DataFrame
import seaborn as sns
import os
os.chdir('C:\\Users\\wuzix\\Desktop\\final big\\archive')
df = spark.read.option('header','true').csv('C:\\Users\\wuzix\\Desktop\\final big\\archive\\movies_metadata.csv',inferSchema=True)
df.show()
```



```
+-----+--------------------+------+--------------------+--------------------+--------------------+--------------------+--------------------+----------
------------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------
+---------+--------------------+--------------------+--------+--------+--------------------+--------------------+------------+--------------------+-----
+-----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
|adult|belongs_to_collection|budget|              genres|            homepage|                  id|             imdb_id|   original_language|
original_title|            overview|          popularity|         poster_path|production_companies|production_countries|        release_date|   revenue|
runtime|    spoken_languages|  status|tagline|               title|               video|vote_average|          vote_count| _c24|
_c25|_c26|_c27|_c28|_c29|_c30|_c31|_c32|_c33|_c34|_c35|_c36|_c37|_c38|_c39|_c40|_c41|_c42|_c43|_c44|
+-----+--------------------+------+--------------------+--------------------+--------------------+--------------------+--------------------+----------
------------------------+--------------------+--------------------+--------------------+--------------------+--------------------+--------------------
+---------+--------------------+--------------------+--------+--------+--------------------+--------------------+------------+--------------------+-----
+-----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+----+
|FALSE|                 null|     0|[{'id': 99, 'name...|http://www.thethi...|               38111|           tt1436475|                  en|
The Third Jihad|The Third Jihad, ...|                null|                null|                null|                null|                null|     null|
null|                null|    null|   null|                null|                null|        null|                null| null|
null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|
|False|                 null|     0|         [{'id': 18"|    'name': 'Drama'}|        {'id': 10751| 'name': 'Family'...|http://www.bbc.co...|
42548|           tt1555126|                  en|The Secret Diarie...|In nineteenth cen...|            2.079269|/nsc5ZLk18MLqAHbV...|       []|
[{'iso_3166_1': '...|            2010/3/1|       0|     90|[{'iso_639_1': 'e...|            Released|        null|The Secret Diarie...|FALSE|  6.2|
10|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|null|
|FALSE|                 null|     0|[{'id': 99, 'name...|http://30for30.es...|               54744|           tt1199479|                  en|
```

```
df.printSchema()
```

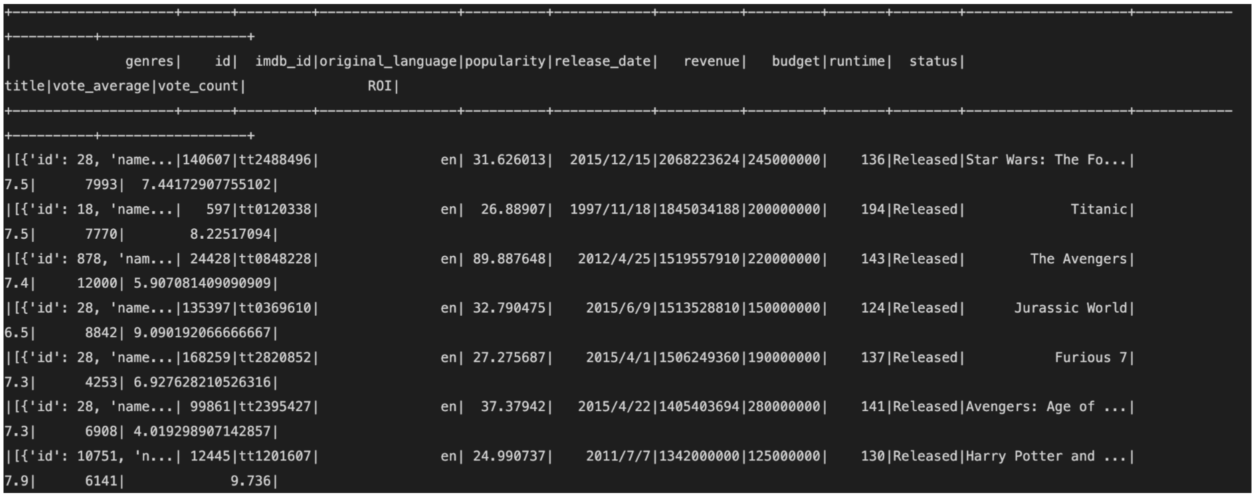


```
root
 |-- adult: string (nullable = true)
 |-- belongs_to_collection: string (nullable = true)
 |-- budget: string (nullable = true)
 |-- genres: string (nullable = true)
 |-- homepage: string (nullable = true)
 |-- id: string (nullable = true)
 |-- imdb_id: string (nullable = true)
 |-- original_language: string (nullable = true)
 |-- original_title: string (nullable = true)
 |-- overview: string (nullable = true)
 |-- popularity: string (nullable = true)
 |-- poster_path: string (nullable = true)
 |-- production_companies: string (nullable = true)
 |-- production_countries: string (nullable = true)
 |-- release_date: string (nullable = true)
 |-- revenue: string (nullable = true)
 |-- runtime: string (nullable = true)
 |-- spoken_languages: string (nullable = true)
 |-- status: string (nullable = true)
 |-- tagline: string (nullable = true)
 |-- title: string (nullable = true)
 |-- video: string (nullable = true)
 |-- vote_average: string (nullable = true)
 |-- vote_count: string (nullable = true)
...
 |-- _c42: string (nullable = true)
 |-- _c43: string (nullable = true)
 |-- _c44: string (nullable = true)
```

```
rating = spark.read.option('header','true').csv('ratings.csv',inferSchema=True)
rating2 = rating.groupby('movieId').agg(F.mean('rating')).select(['movieId','avg(rating)'])
df1 =rating2.join(df, df.id==rating2.movieId, how='inner')
df.corr('popularity', 'runtime')
df.corr('popularity', 'vote_count')
df.corr('runtime', 'vote_count')
```

```
df1 = df1.filter((df.revenue > 1000000)&(df.budget > 100000)&(df.runtime >
0) ).select(['genres','id','imdb_id','original_language','popularity', 'release_date', 'revenue', 'budget',
'runtime','status', 'title','vote_average' , 'vote_count'])
```

```
from pyspark.sql import functions as F
df2=df1.withColumn('ROI', (F.col('revenue')-F.col('budget'))/F.col('budget'))
df2.show()
```



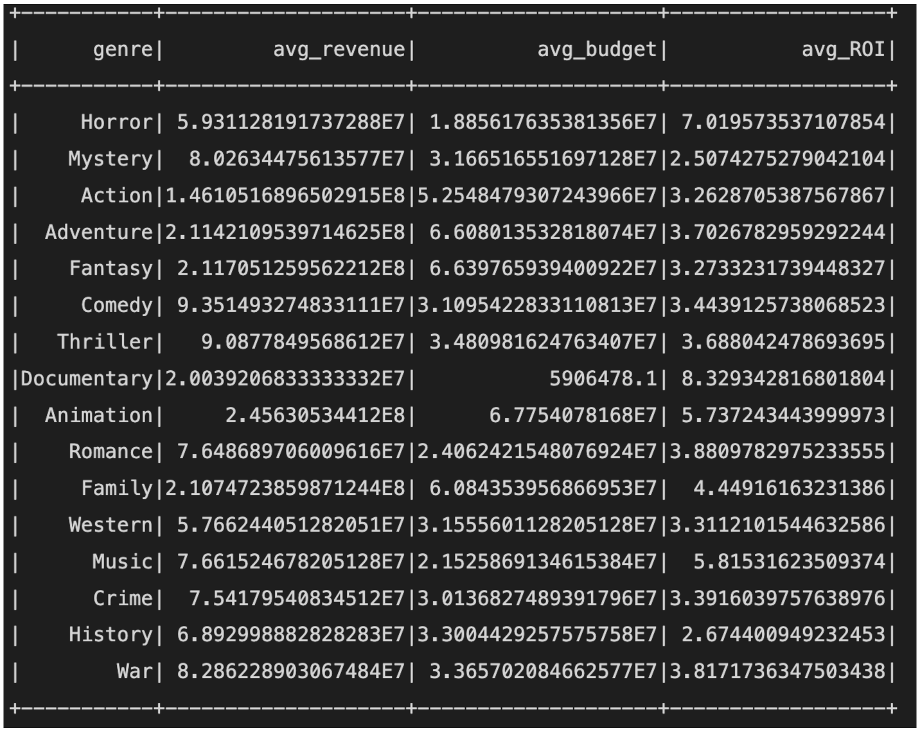
```
+--------------------+------+---------+-----------------+----------+------------+----------+---------+-------+--------+--------------------+------------
+----------+------------------+
|              genres|    id|  imdb_id|original_language|popularity|release_date|   revenue|   budget|runtime|  status|
title|vote_average|vote_count|               ROI|
+--------------------+------+---------+-----------------+----------+------------+----------+---------+-------+--------+--------------------+------------
+----------+------------------+
|[{'id': 28, 'name...|140607|tt2488496|               en| 31.626013|  2015/12/15|2068223624|245000000|    136|Released|Star Wars: The Fo...|
7.5|      7993|  7.44172907755102|
|[{'id': 18, 'name...|   597|tt0120338|               en|  26.88907|  1997/11/18|1845034188|200000000|    194|Released|             Titanic|
7.5|      7770|        8.22517094|
|[{'id': 878, 'nam...| 24428|tt0848228|               en| 89.887648|   2012/4/25|1519557910|220000000|    143|Released|        The Avengers|
7.4|     12000| 5.907081409090909|
|[{'id': 28, 'name...|135397|tt0369610|               en| 32.790475|    2015/6/9|1513528810|150000000|    124|Released|      Jurassic World|
6.5|      8842| 9.090192066666667|
|[{'id': 28, 'name...|168259|tt2820852|               en| 27.275687|    2015/4/1|1506249360|190000000|    137|Released|           Furious 7|
7.3|      4253| 6.927628210526316|
|[{'id': 28, 'name...| 99861|tt2395427|               en|  37.37942|   2015/4/22|1405403694|280000000|    141|Released|Avengers: Age of ...|
7.3|      6908| 4.019298907142857|
|[{'id': 10751, 'n...| 12445|tt1201607|               en| 24.990737|    2011/7/7|1342000000|125000000|    130|Released|Harry Potter and ...|
7.9|      6141|             9.736|
```

Genre - Finance

```
genre = ['Horror', 'Mystery', 'Action', 'Adventure', 'Fantasy', 'Comedy', 'Thriller', 'Documentary',
'animation', 'romance', 'family', 'western','music' , 'crime', 'history', 'war']
genre = [s.capitalize() for s in genre]
for name in genre:
    locals()[name] = df2.filter(F.col("genres").contains(name))
```

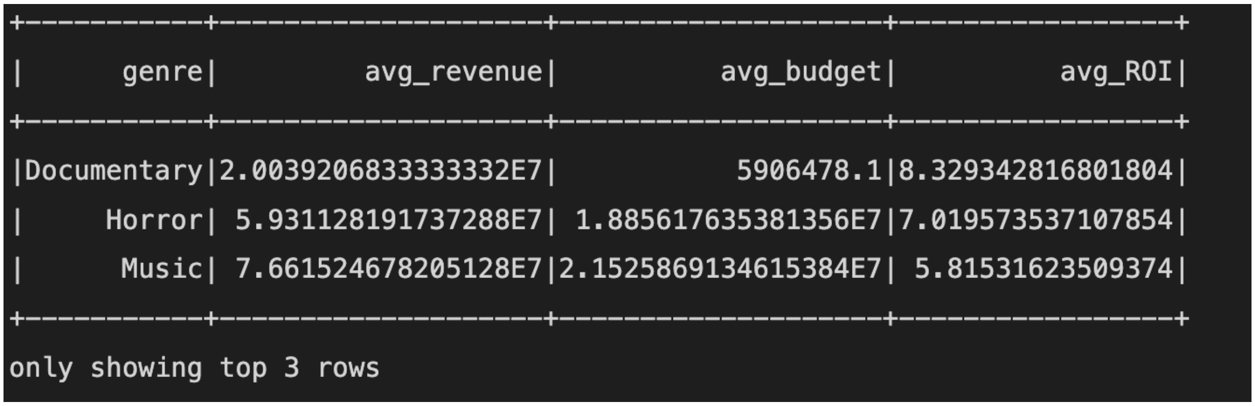
```
from pyspark.sql import SparkSession
from pyspark.sql import SQLContext
sqlContext = SQLContext(sc)
```

```
data = []
for name in genre:
        data.append((name,locals()[name].agg(F.mean('revenue')).collect()[0][0],locals()[name].agg(F.m
ean('budget')).collect()[0][0],locals()[name].agg(F.mean('ROI')).collect()[0][0]))
genre_fin_df = sqlContext.createDataFrame(data, ('genre', 'avg_revenue', 'avg_budget', 'avg_ROI'))
genre_fin_df.show()
```



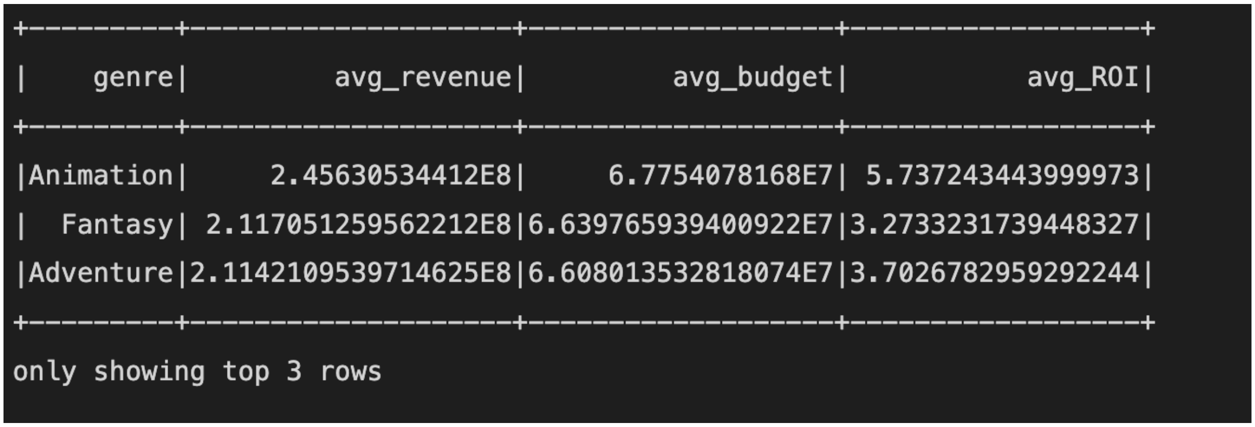
```
+-----------+--------------------+--------------------+------------------+
|      genre|         avg_revenue|          avg_budget|           avg_ROI|
+-----------+--------------------+--------------------+------------------+
|     Horror| 5.931128191737288E7| 1.885617635381356E7| 7.019573537107854|
|    Mystery|  8.02634475613577E7| 3.166516551697128E7|2.5074275279042104|
|     Action|1.4610516896502915E8|5.2548479307243966E7|3.2628705387567867|
|  Adventure|2.1142109539714625E8| 6.608013532818074E7|3.7026782959292244|
|    Fantasy| 2.117051259562212E8| 6.639765939400922E7|3.2733231739448327|
|     Comedy| 9.351493274833111E7|3.1095422833110813E7|3.4439125738068523|
|   Thriller|   9.0877849568612E7| 3.480981624763407E7| 3.688042478693695|
|Documentary|2.0039206833333332E7|           5906478.1| 8.329342816801804|
|  Animation|     2.45630534412E8|      6.7754078168E7| 5.737243443999973|
|    Romance| 7.648689706009616E7|2.4062421548076924E7|3.8809782975233555|
|     Family|2.1074723859871244E8| 6.084353956866953E7|  4.44916163231386|
|    Western| 5.766244051282051E7|3.1555601128205128E7|3.3112101544632586|
|      Music| 7.661524678205128E7|2.1525869134615384E7|  5.81531623509374|
|      Crime|  7.54179540834512E7|3.0136827489391796E7|3.3916039757638976|
|    History| 6.892998882828283E7|3.3004429257575758E7| 2.674400949232453|
|        War| 8.286228903067484E7| 3.365702084662577E7|3.8171736347503438|
+-----------+--------------------+--------------------+------------------+
```

genre_fin_df.orderBy(genre_fin_df.avg_ROI.desc()).show(3)



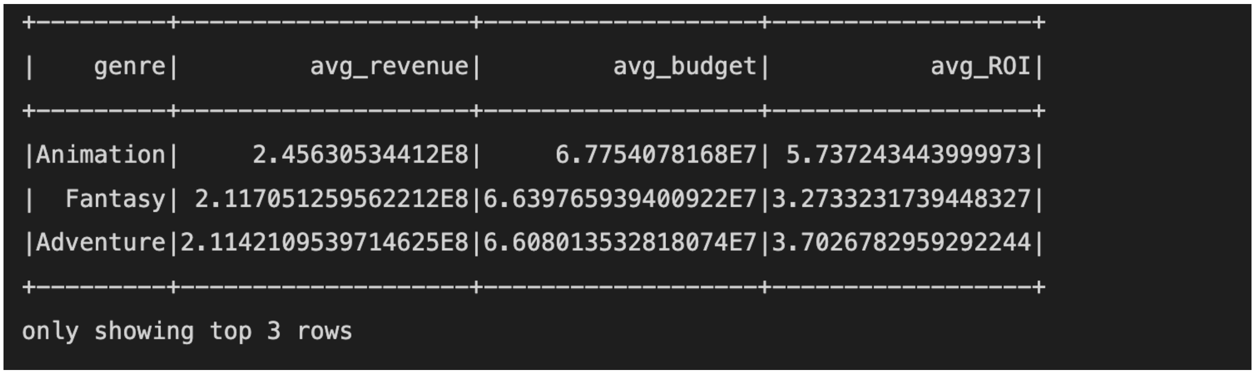
```
+-----------+--------------------+--------------------+-----------------+
|      genre|         avg_revenue|          avg_budget|          avg_ROI|
+-----------+--------------------+--------------------+-----------------+
|Documentary|2.0039206833333332E7|           5906478.1|8.329342816801804|
|     Horror| 5.931128191737288E7| 1.885617635381356E7|7.019573537107854|
|      Music| 7.661524678205128E7|2.1525869134615384E7| 5.81531623509374|
+-----------+--------------------+--------------------+-----------------+
only showing top 3 rows
```

genre_fin_df.orderBy(genre_fin_df.avg_revenue.desc()).show(3)



```
+---------+--------------------+-------------------+------------------+
|    genre|         avg_revenue|         avg_budget|           avg_ROI|
+---------+--------------------+-------------------+------------------+
|Animation|     2.45630534412E8|     6.7754078168E7| 5.737243443999973|
|  Fantasy| 2.117051259562212E8|6.639765939400922E7|3.2733231739448327|
|Adventure|2.1142109539714625E8|6.608013532818074E7|3.7026782959292244|
+---------+--------------------+-------------------+------------------+
only showing top 3 rows
```

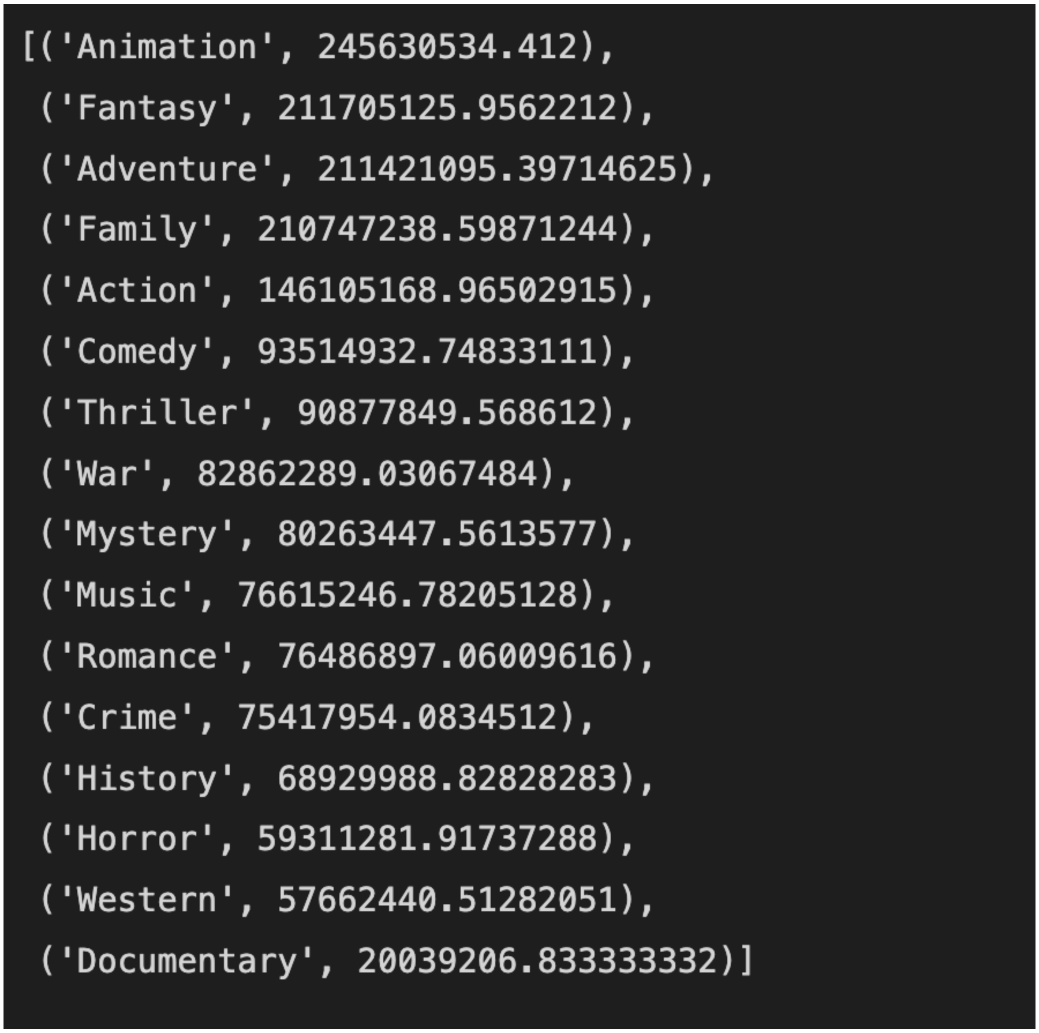
genre_fin_df.orderBy(genre_fin_df.avg_budget.desc()).show(3)



```
+---------+--------------------+-------------------+------------------+
|    genre|         avg_revenue|         avg_budget|           avg_ROI|
+---------+--------------------+-------------------+------------------+
|Animation|     2.45630534412E8|     6.7754078168E7| 5.737243443999973|
|  Fantasy| 2.117051259562212E8|6.639765939400922E7|3.2733231739448327|
|Adventure|2.1142109539714625E8|6.608013532818074E7|3.7026782959292244|
+---------+--------------------+-------------------+------------------+
only showing top 3 rows
```

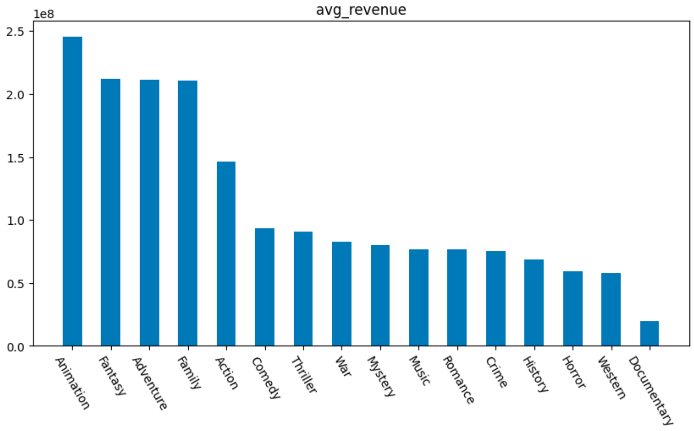
rev_data = sorted(dict(zip([genre_fin_df.collect()[i][0] for i in range(16)],[genre_fin_df.collect()[i][1] for i in range(16)])).items(), key = lambda x:x[1],reverse = True)
bud_data = sorted(dict(zip([genre_fin_df.collect()[i][0] for i in range(16)],[genre_fin_df.collect()[i][2] for i in range(16)])).items(), key = lambda x:x[1],reverse = True)
ROI_data = sorted(dict(zip([genre_fin_df.collect()[i][0] for i in range(16)],[genre_fin_df.collect()[i][3] for i in range(16)])).items(), key = lambda x:x[1],reverse = True)

rev_data

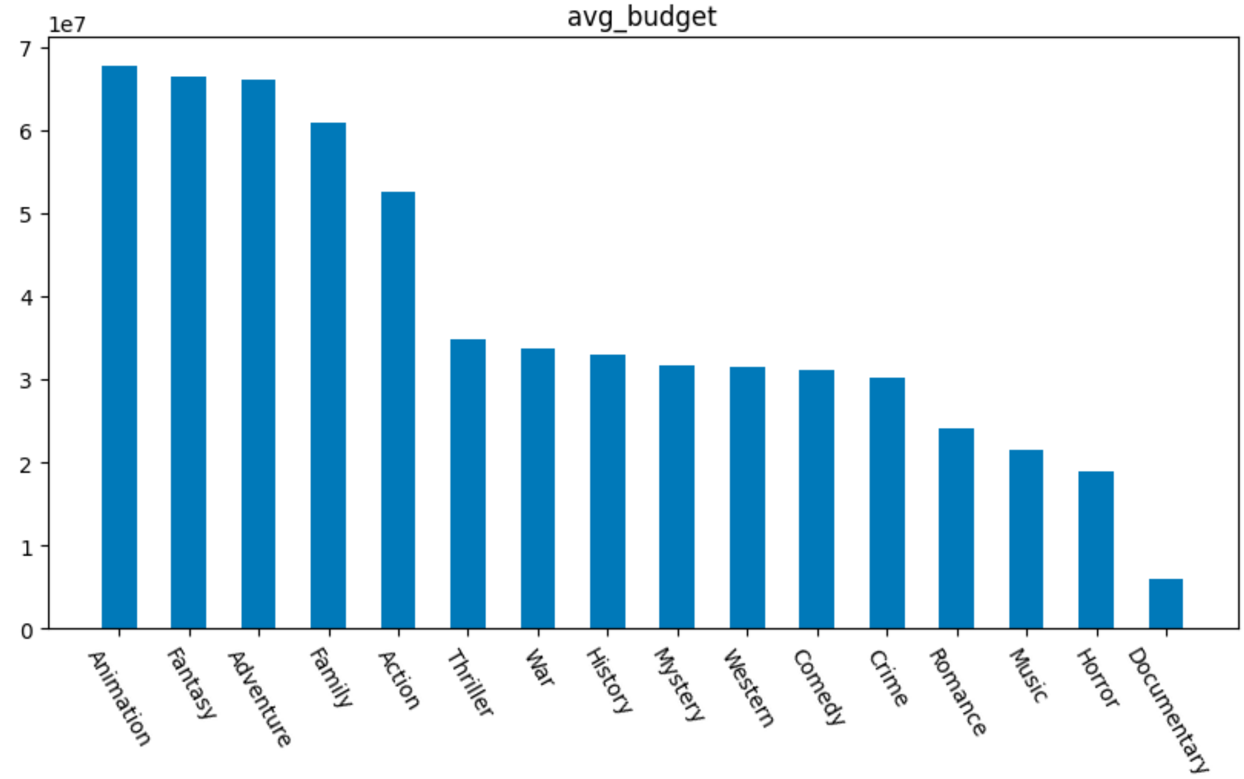


```
[('Animation', 245630534.412),
 ('Fantasy', 211705125.9562212),
 ('Adventure', 211421095.39714625),
 ('Family', 210747238.59871244),
 ('Action', 146105168.96502915),
 ('Comedy', 93514932.74833111),
 ('Thriller', 90877849.568612),
 ('War', 82862289.03067484),
 ('Mystery', 80263447.5613577),
 ('Music', 76615246.78205128),
 ('Romance', 76486897.06009616),
 ('Crime', 75417954.0834512),
 ('History', 68929988.82828283),
 ('Horror', 59311281.91737288),
 ('Western', 57662440.51282051),
 ('Documentary', 20039206.833333332)]
```

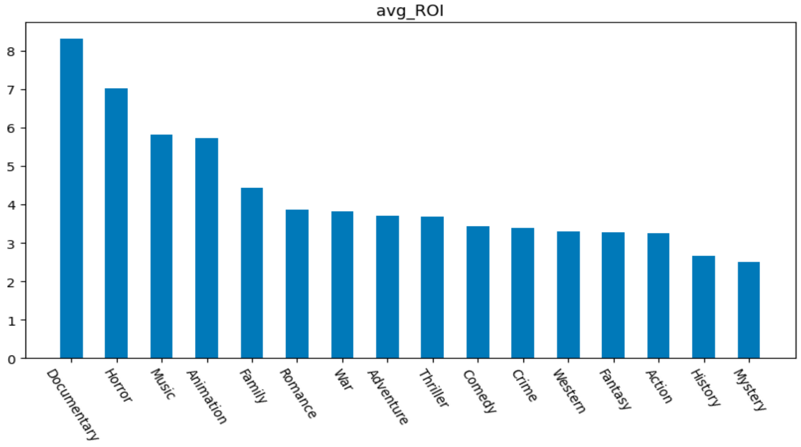
```
plt.figure(figsize=(10,5))
plt.bar(range(len(rev_data)),[i[1] for i in rev_data], tick_label=[i[0] for i in rev_data], width= 0.5)
plt.title('avg_revenue')
plt.xticks(rotation=300)
plt.show()
```



```
plt.figure(figsize=(10,5))
plt.bar(range(len(bud_data)),[i[1] for i in bud_data], tick_label=[i[0] for i in bud_data], width= 0.5)
plt.title('avg_budget')
plt.xticks(rotation=300)
plt.show()
```

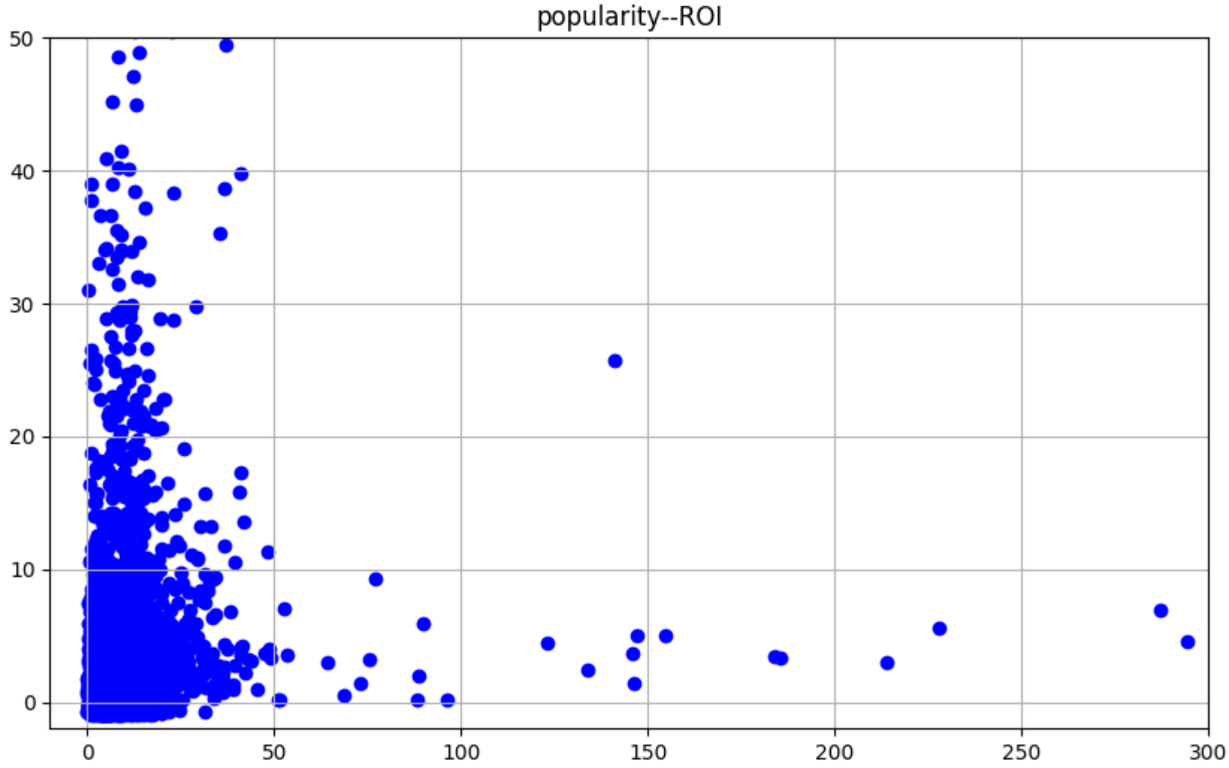


```
plt.figure(figsize=(10,5))
plt.bar(range(len(ROI_data)),[i[1] for i in ROI_data], tick_label=[i[0] for i in ROI_data], width= 0.5)
plt.title('avg_ROI')
plt.xticks(rotation=300)
plt.show()
```



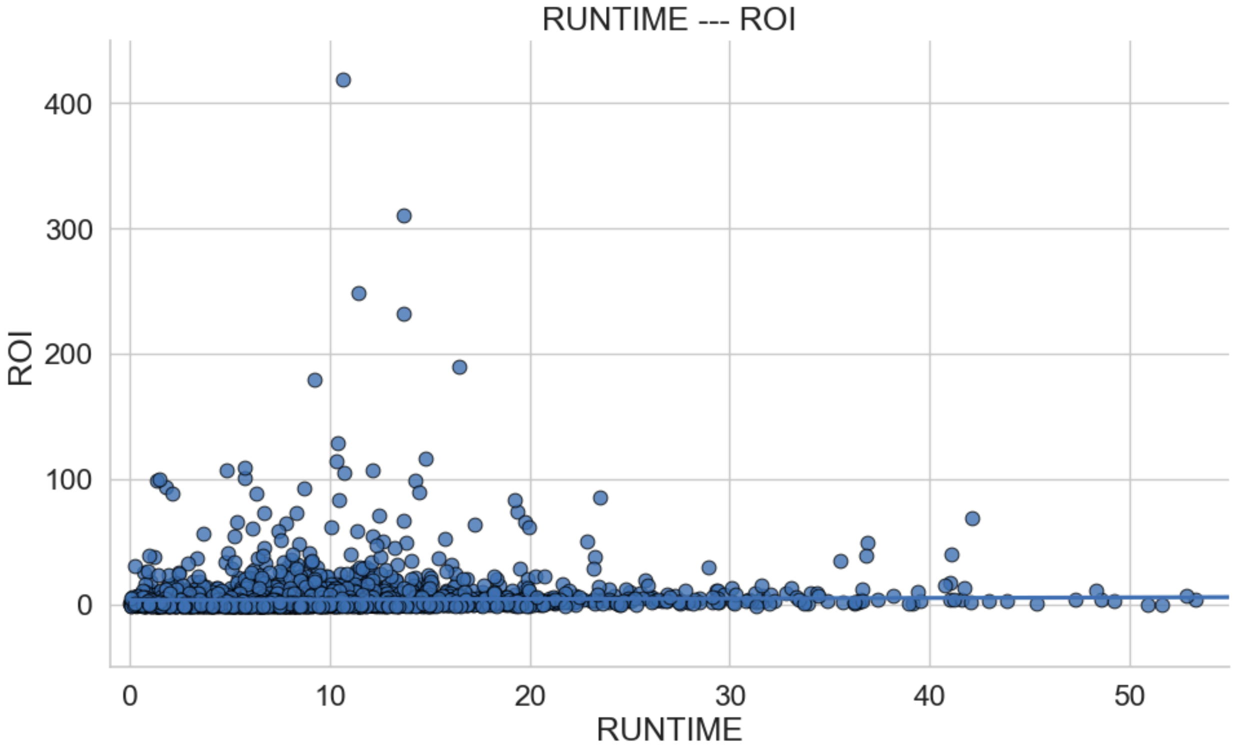
Popularity - Finance

```
df_new = df2.withColumn("popularity",df2.popularity.cast('double'))
df_new = df_new.withColumn("runtime",df_new.runtime.cast('double'))
df_new = df_new.withColumn("vote_average",df_new.vote_average.cast('double'))
df_new = df_new.withColumn("vote_count",df_new.vote_count.cast('double'))

pop_data = [df_new.collect()[i][4] for i in range(4359)]
ROI_row_data = [df_new.collect()[i][13] for i in range(4359)]
plt.figure(figsize = (10,6))
plt.scatter(pop_data, ROI_row_data ,color="blue")
plt.xlim((-10, 300))
plt.ylim((-2, 50))
plt.title('popularity--ROI')
plt.grid()
plt.show()
```

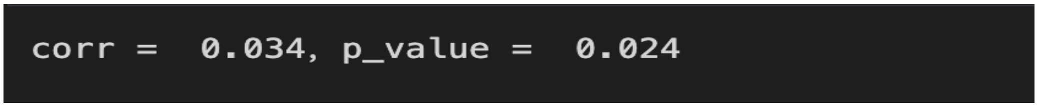


```
a={'ROI':ROI_row_data, 'RUNTIME':pop_data}
data1=DataFrame(a)
g1 = sns.lmplot(data =
data1,x='RUNTIME',y='ROI',height=7,aspect=1.6,palette='Set1',scatter_kws=dict(s=60, linewidths=.7,
edgecolors='black'))
sns.set(style="whitegrid", font_scale=1.5)
g1.set(xlim=(-1, 55), ylim=(-50, 450))
```

```
g1.fig.set_size_inches(10, 6)
g1.tight_layout()
plt.title("RUNTIME --- ROI")
plt.show()
```



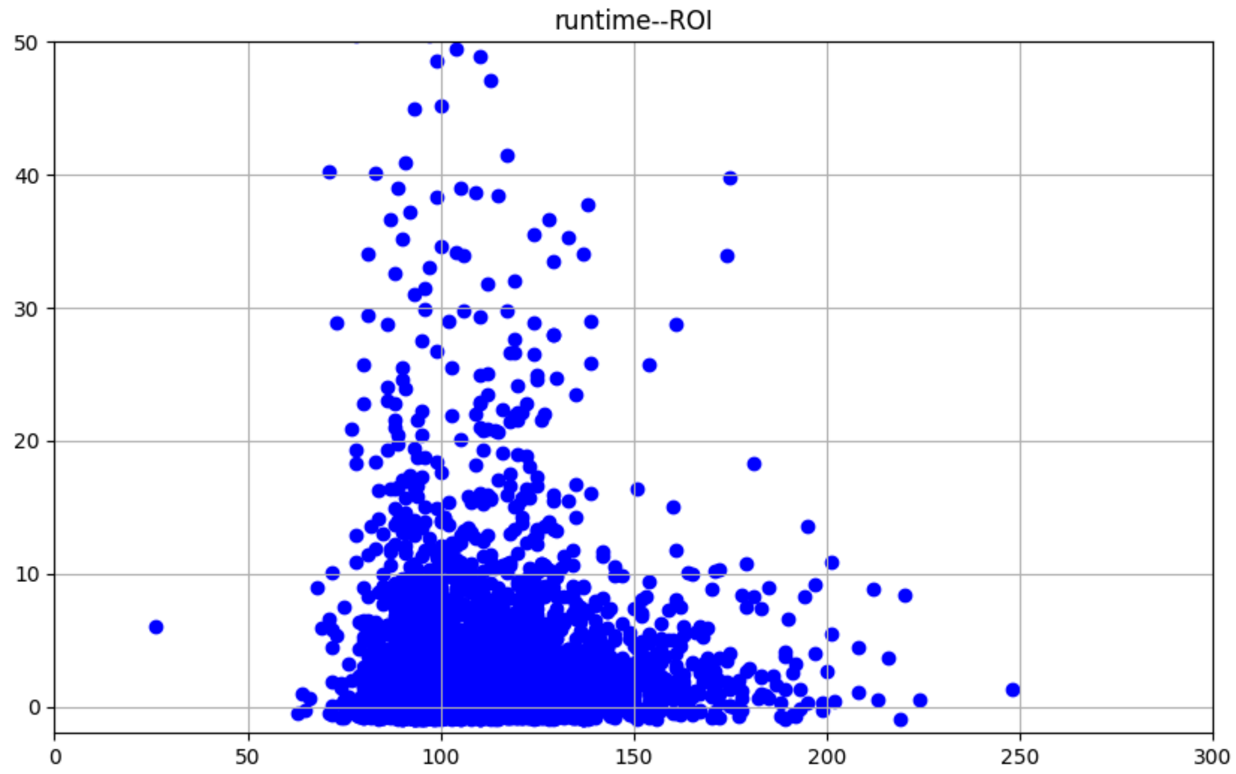
```
import scipy.stats as stats

r,p = stats.pearsonr(ROI_row_data,pop_data)
print('corr = %6.3f,  p_value = %6.3f'%(r,p))
```



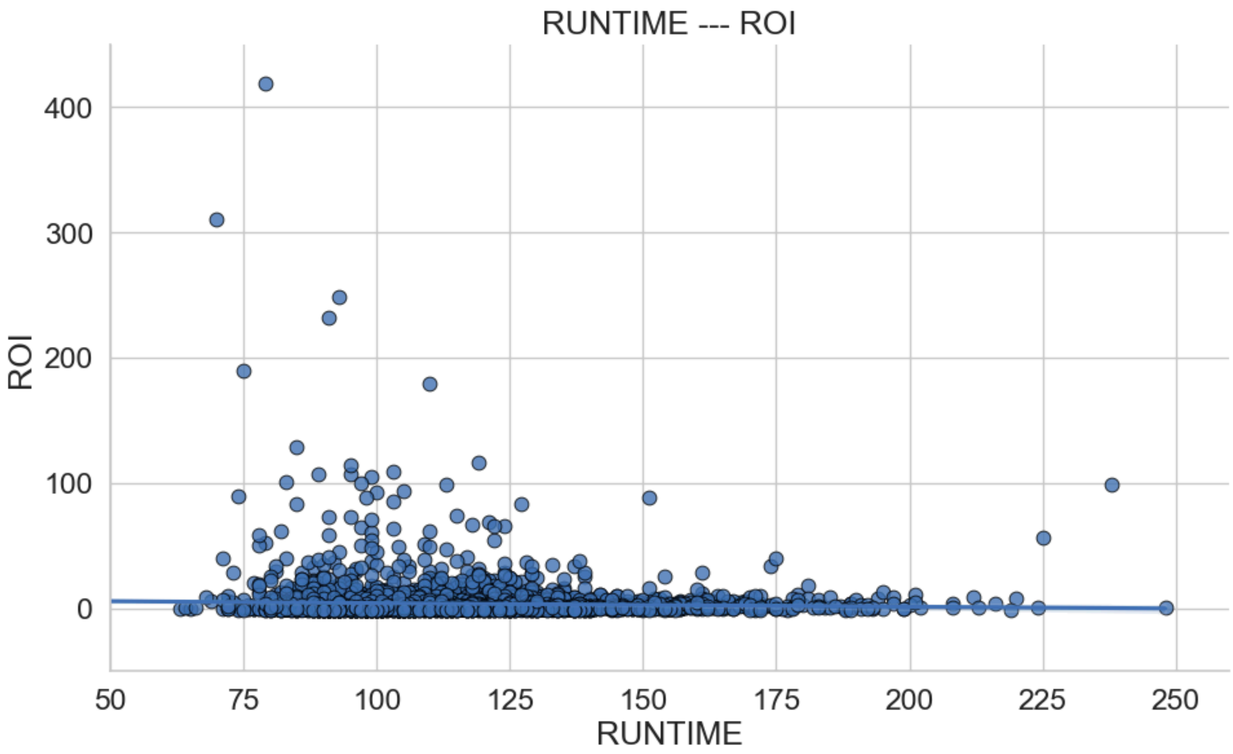
```
corr =  0.034, p_value =  0.024
```

Runtime - Finance

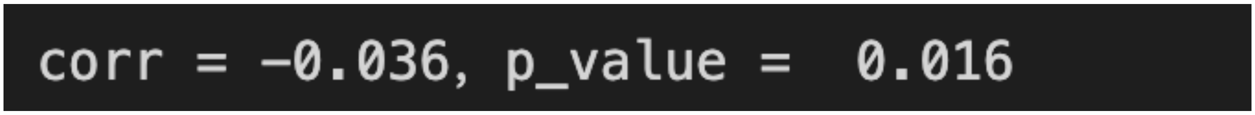
```
runtime_data = [df_new.collect()[i][8] for i in range(4359)]
plt.figure(figsize = (10,6))
plt.scatter(runtime_data, ROI_row_data,color="blue")
plt.xlim((0, 300))
plt.ylim((-2, 50))
plt.title('runtime--ROI')
plt.grid()
plt.show()
```



```
b={'ROI':ROI_row_data, 'RUNTIME':runtime_data}
data2=DataFrame(b)
g2 = sns.lmplot(data =
data2,x='RUNTIME',y='ROI',height=7,aspect=1.6,palette='Set1',scatter_kws=dict(s=60, linewidths=.7,
edgecolors='black'))
sns.set(style="whitegrid", font_scale=1.5)
g2.set(xlim=(50, 260), ylim=(-50, 450))
g2.fig.set_size_inches(10, 6)
g2.tight_layout()
plt.title("RUNTIME --- ROI")
plt.show()
```



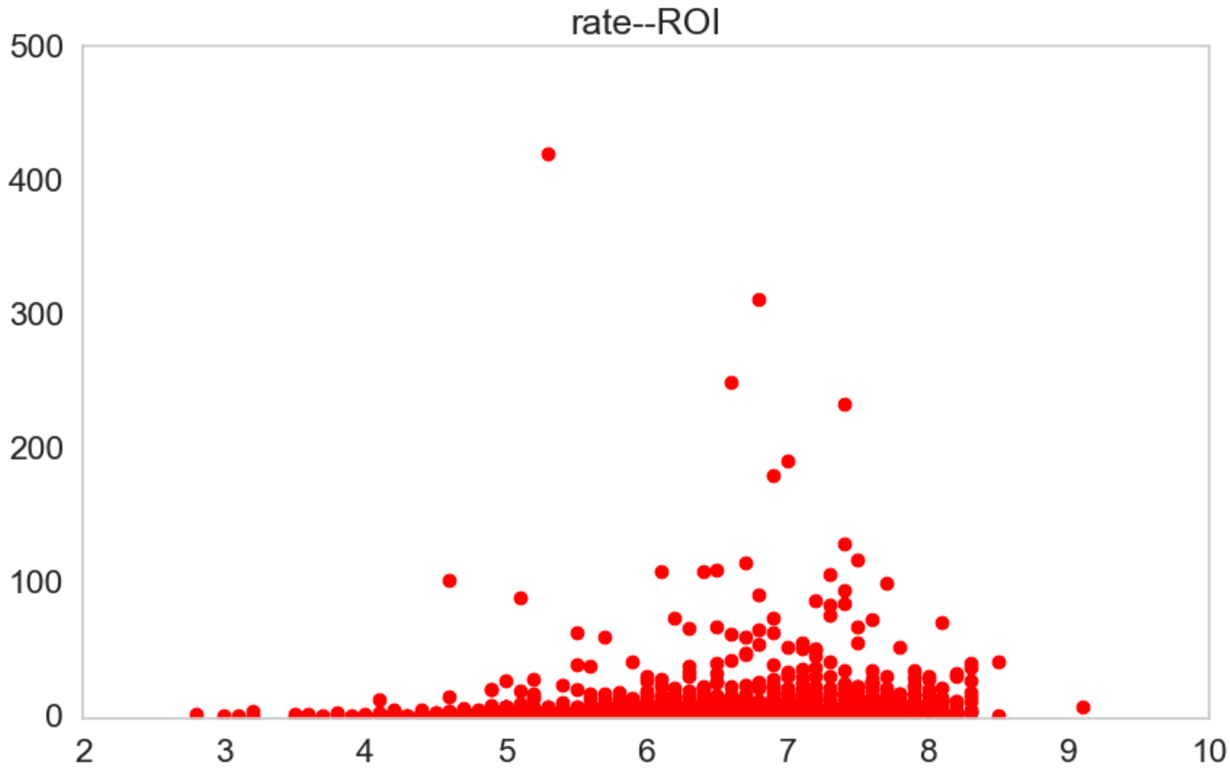
```
import scipy.stats as stats

r,ru = stats.pearsonr(ROI_row_data,runtime_data)
print('corr = %6.3f, p_value = %6.3f'%(r,ru))
```



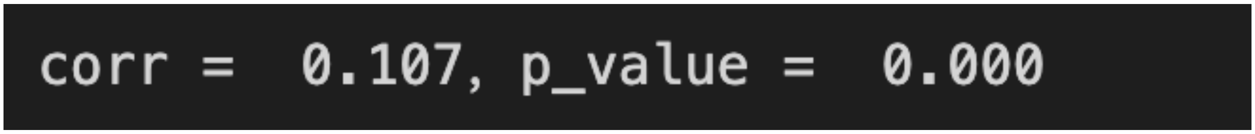
```
corr = -0.036, p_value =  0.016
```

Rate - Finance

```
df_vote = df_new.filter(df_new.vote_count > 100)
rate_data = [df_vote.collect()[i][11] for i in range(3321)]
ROI_row_data1 = [df_vote.collect()[i][13] for i in range(3321)]
plt.figure(figsize = (10,6))
plt.scatter(rate_data, ROI_row_data1,color="Red")
plt.xlim((2, 10))
plt.ylim((-2, 500))
plt.title('rate--ROI')
plt.grid()
plt.show()
```

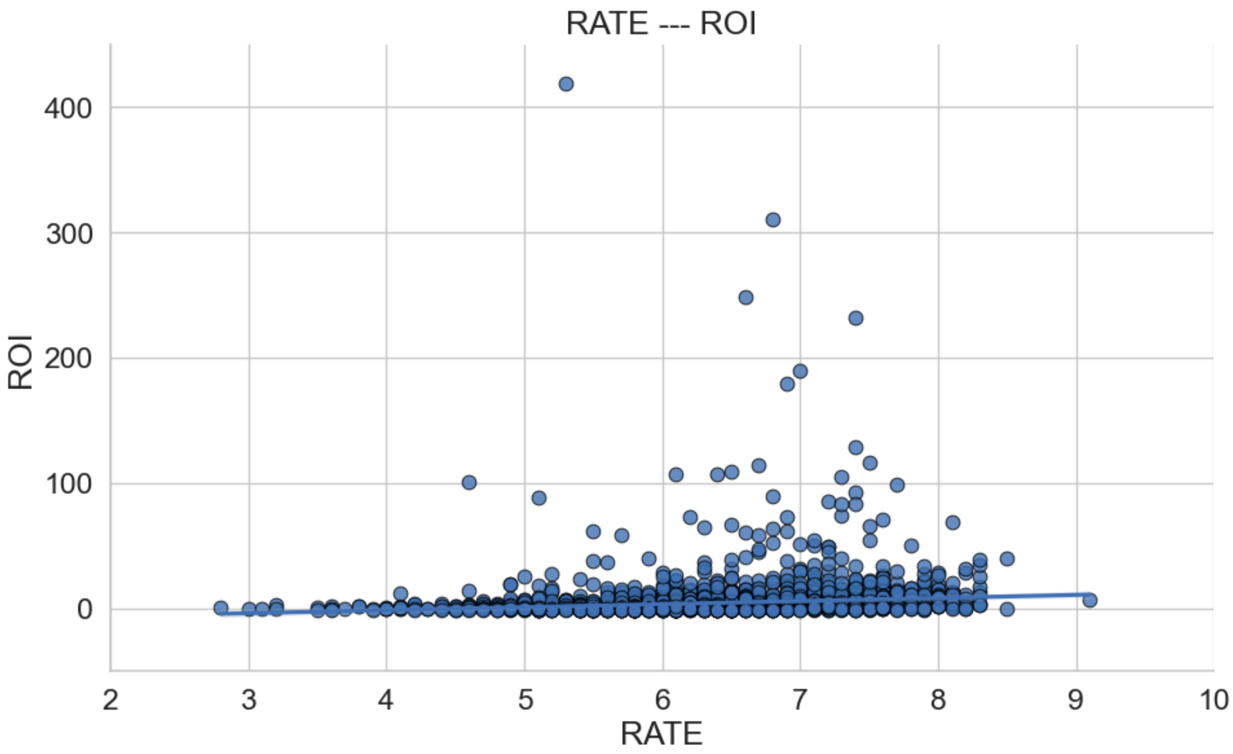


```
import scipy.stats as stats

r1,ra = stats.pearsonr(ROI_row_data1,rate_data)
print('corr = %6.3f,  p_value = %6.3f'%(r1,ra))
```



```
corr =  0.107, p_value =  0.000
```

```
c={'ROI':ROI_row_data1, 'RATE':rate_data}
data3=DataFrame(c)
g3 = sns.lmplot(data =
data3,x='RATE',y='ROI',height=7,aspect=1.6,palette='Set1',scatter_kws=dict(s=60, linewidths=.7,
edgecolors='black'))
sns.set(style="whitegrid", font_scale=1.5)
g3.set(xlim=(2, 10), ylim=(-50, 450))
g3.fig.set_size_inches(10, 6)
g3.tight_layout()
plt.title("RATE --- ROI")
plt.show()
```


RATE --- ROI
ROI
400
300
200
100
0
2
3
4
5
6
7
8
9
10
RATE